Diskreter Logarithmus

Josef Schmeißer und Fabian Grotz

18.05.2016

1 Motivation

2 Gruppentheorie
- Primitivwurzel
- Kleiner fermatscher Satz
- Diskretes Logarithmus-Problem
- Logarithmusgesetze

3 Algorithmen zur Bestimmung des diskreten Logarithmus
- Index-Calculus
- Babystep-Giantstep-Algorithmus

4 Anwendungen in der Kryptographie
- Elgamal-Verschlüsselungsverfahren
- Massey-Omura-Schema

5 Ausblick

Gründe den diskreten Logarithmus näher zu betrachten u.a.:

- Grundlage einiger asymmetrischer Kryptosysteme
- Alternative zum Faktorisierungsproblem
- Basis einiger Schlüsselaustauschprotokolle

Sei $(\mathbb{G}, \odot)$ eine Gruppe, wir definieren:

Definition

- e bezeichnet das neutrale Element
- $ord(\mathbb{G}) := |\mathbb{G}|$
- Für $\alpha \in \mathbb{G}$ ist $ord(\alpha) = n$ mit $\alpha^n = e$

Definition

Eine Gruppe $\mathbb{G}$ heißt zyklisch, wenn ein $g \in \mathbb{G}$ existiert, so dass:

$$\forall \alpha \in \mathbb{G} : \exists i \in \mathbb{N} : g^i = \alpha$$

Wir nennen g einen Generator der zyklischen Gruppe.

- Sei $(\mathbb{G}, \odot)$ eine Gruppe und $\alpha \in \mathbb{G}$.
- α sei von endlicher Ordnung.

Definition

$\langle\alpha\rangle$ bezeichnet die von α erzeugte Untergruppe.

Die Euler'sche $\varphi(n)$-Funktion ist wie folgt definiert:

Definition

Sie gibt für eine natürliche Zahl n an, wie viele zu n teilerfremde natürliche Zahlen existieren, welche nicht größer als n sind:

$$\varphi(n) \ := \ \left|\{a \in \mathbb{N} \,|\, 1 \leq a \leq n \wedge ggT(a,n) = 1\}\right|$$

Sei $\mathbb{G}$ die prime Restklassengruppe $(\mathbb{Z}/p\mathbb{Z})^{\times}$ mit der Multiplikation als vorherrschende Operation (gekennzeichnet durch $\times$).

Definition

Ein Element $\alpha \in \mathbb{G}$ ist eine Primitivwurzel modulo p, wenn gilt:

$$ord(\alpha) = \varphi(p)$$

Satz (Lagrange)

Sei $\mathbb{G}$ eine endliche Gruppe und $\mathbb{H} \subset \mathbb{G}$ eine Untergruppe. Dann ist $ord(\mathbb{H})$ ein Teiler von $ord(\mathbb{G})$.

Satz (Lagrange)

Sei $\mathbb{G}$ eine endliche Gruppe und $\mathbb{H} \subset \mathbb{G}$ eine Untergruppe. Dann ist $ord(\mathbb{H})$ ein Teiler von $ord(\mathbb{G})$.

Satz (Fermat)

Sei p eine Primzahl. Für jede nicht durch p teilbare ganze Zahl a gilt:

$$a^{p-1} \equiv 1 \; mod \; p$$

Nachfolgend sei $(\mathbb{G}, \odot)$ eine zyklische Gruppe mit Erzeuger g und $n := ord(\mathbb{G})$.

Definition

Die diskrete Exponentialfunktion ist gegeben durch:

$$\begin{aligned} exp_g : (\mathbb{Z}/n\mathbb{Z}, +) &\to \mathbb{G} \\ k &\mapsto g^k \in \mathbb{G} \end{aligned}$$

Definition

Der diskrete Logarithmus bestimmt das Urbild von exp_g:

$$\begin{aligned} log_g : \mathbb{G} &\to (\mathbb{Z}/n\mathbb{Z}, +) \\ x &\mapsto log_g(x),\ \ mit\ exp_g(log_g(x)) = x \end{aligned}$$

Definition

Der diskrete Logarithmus bestimmt das Urbild von exp_g:

$$\begin{aligned} log_g : \mathbb{G} &\rightarrow (\mathbb{Z}/n\mathbb{Z}, +) \\ x &\mapsto log_g(x), \ \ mit\ exp_g(log_g(x)) = x \end{aligned}$$

Beispiel

Sei $\mathbb{G} = (\mathbb{Z}/13\mathbb{Z})^\times$ und $g = 2$ eine Primitivwurzel modulo 13:

x	1	2	3	4	5	6	7	8	9	10	11	12
$\log_2\ x$	0	1	4	2	9	5	11	3	8	10	7	6

Die Logarithmusgesetze gelten auch für den diskreten Logarithmus.

Satz

Sei $(\mathbb{G}, \odot)$ eine zyklische Gruppe, $g \in \mathbb{G}$, $k \in \mathbb{Z}$ und $x, y \in \langle g \rangle$, dann gilt:

- $log_g(x \odot y) = log_g(x) + log_g(y)$
- $log_g(x^k) = k \cdot log_g(x)$

Index-Calculus

Eingabe: Eine multiplikative Gruppe $(\mathbb{Z}/p\mathbb{Z})^\times$,
ein Generator g und ein $\alpha \in (\mathbb{Z}/p\mathbb{Z})^\times$.
Ausgabe: Der diskrete Logarithmus $x = log_g(\alpha)$

Der Algorithmus teil sich in zwei Phasen:

1. Sammeln von Relationen und lösen eines GLS
2. Berechnung des diskreten Logarithmus für α

Phase 1:

1. Bestimme die Menge $B = \{p_1, ..., p_k\}$ aller Primzahlen bis p_k mit $p_k < p$.
2. Wähle $l \geq k$ zufällige $x_1, ..., x_l \in (\mathbb{Z}/(p-1)\mathbb{Z}, +)$ mit:
   - x_i und x_j paarweise verschieden für alle $i, j \in \{1, ..., l\}$ und
   - g^{x_i} *mod* p ist mit B faktorisierbar

Wir erhalten l Gleichungen:

$$g^{x_1} = \prod_{i=1}^{k} p_i^{e_{1,i}} \; mod \; p$$

$$\vdots$$

$$g^{x_l} = \prod_{i=1}^{k} p_i^{e_{l,i}} \; mod \; p$$

Logarithmieren liefert uns lineare Gleichungen:

$$x_1 = \sum_{i=1}^{k} e_{1,i} \cdot log_g\ p_i\ mod\ (p-1)$$

$$\vdots$$

$$x_l = \sum_{i=1}^{k} e_{l,i} \cdot log_g\ p_i\ mod\ (p-1)$$

Phase 2:

1. Wähle ein zufälliges $x \in (\mathbb{Z}/(p-1)\mathbb{Z}, +)$, sodass: g^x *mod* p mit B faktorisierbar ist.
2. Mit Hilfe der $l+1$ Gleichungen werden die $log_g\, p_i$ für $i \in \{1, ..., k\}$ und $log_g\, \alpha$ bestimmt.

Es gilt:

$$g^x \odot \alpha = \prod_{i=1}^{k} p_i^{e_i} \ mod \ p$$

$$\Longrightarrow \ log_g \alpha = -x + \sum_{i=1}^{k} e_i \cdot log_g p_i \ mod \ (p-1)$$

Babystep-Giantstep-Algorithmus

Theorie

- Gegeben Zyklische Gruppe $\mathbb{G}$ mit Ordnung n, Generator g und ein Element der Gruppe α
- Gesucht ist x sodass $g^x = \alpha$
- setzen $x = i \cdot m + j$
- m sollte in $\lceil \sqrt{n} \rceil$ sein
- ausserdem $0 \leq i < m$ und $0 \leq j < m$

Theorie

$$g^{im+j} = \alpha \Leftrightarrow g^j = a(g^{-m})^i$$

Theorie

$$g^{im+j} = \alpha \Leftrightarrow g^j = a(g^{-m})^i$$

Babystep

Berechne für alle j den Ausdruck g^j. Paare (j, g^j) werden in Tabelle gespeichert

Theorie

$$g^{im+j} = \alpha \Leftrightarrow g^j = a(g^{-m})^i$$

Babystep

Berechne für alle j den Ausdruck g^j. Paare (j, g^j) werden in Tabelle gespeichert

Giantstep

Berechne $(g^{-m})^i$ und vergleiche mit Tabelle. Wenn Treffer, gib $x = im + j$ aus.

Algorithmus in Pseudocode

Eingabe

zyklische Gruppe G der Ordnung n mit einem Generator g und ein Element der Gruppe α

Berechnung

1. Setze $m := \lceil \sqrt{n} \rceil$
2. Für alle $j \in \{0, \cdots, m-1\}$:
    1. Berechne g^j und speichere das Tupel (j, g^j) in einer Tabelle
3. Setze $t := \alpha$
4. Für alle $i \in \{0, \cdots, m-1\}$:
    1. Suche in der Tabelle nach einem Paar mit $t = g^j$
    2. Wenn Paar existiert gib $im + j = log_g(\alpha)$ aus
    3. Wenn nicht: Setze $t := t * g^{-m}$ und fahre fort

Beispiel

- Wir nehmen eine Gruppe G der Ordnung $n = 29$ mit Erzeuger $g = 11$
- Wir wollen den diskreten Logarithmus von $a = 3$ zur Basis g berechnen, also die Lösung von $3 = 11^x \bmod 29$
- Rechnung siehe Tafel

Beispiel

- Wir nehmen eine Gruppe G der Ordnung $n = 29$ mit Erzeuger $g = 11$
- Wir wollen den diskreten Logarithmus von $a = 3$ zur Basis g berechnen, also die Lösung von $3 = 11^x \bmod 29$
- Rechnung siehe Tafel
- Lösung: $log_{11}3 = 2 \cdot 6 + 5 = 17$

Laufzeit

Laufzeit

hängt von Liste ab, die m-Einträge nach Babystep Berechnungen hat und durchsucht werden muss.
Mit Hashfunktionen kann laufzeit gemindert werden.
Dadurch gesamte Laufzeit bei $O(m) bzw. O(\sqrt{n})$

Speicherverbrauch

Speicherverbrauch

hängt von Liste ab, die m-Einträge nach Babystep Berechnungen hat. Dadurch gesamter Speicherbedarf bei $O(m) bzw. O(\sqrt{n})$

Elgamal-Verschlüsselungsverfahren

Elgamal-Verschlüsselungsverfahren

- entwickelt 1985 von Taher Elgamal
- ist ein Public-Key Verschlüsselungsverfahren
- beruht auf Operationen in einer zyklischen Gruppe endlicher Ordnung

Vorbereitungen

Der Empfänger

1. wählt eine endliche, zyklische Gruppe G der Ordnung n mit Erzeuger p
2. wählt eine zufällige Zahl $a \in \{1, \ldots, n-1\}$ *mit* $ggT(a, n) = 1$ als privater Schlüssel des Empfängers
3. berechnet das Gruppenelement $A = p^a \in G$ als öffentlicher Schlüssel
4. veröffentlicht (G, p) und A

Verschlüsseln

Der Sender

1. möchte Nachricht $m \in G$ versenden
2. wählt $r \in \{1, \ldots, n-1\}$ $mit\ ggT(r, n) = 1$
3. berechnet $R = p^r \in G$
4. berechnet $c = A^r \cdot n \in G$
5. sendet (R,c) an den Empfänger

Entschlüsseln

Der Empfänger

- berechnet $m = \ R^{-a} \cdot c \in G$

Entschlüsseln

Der Empfänger

- berechnet $m = R^{-a} \cdot c \in G$

Es gilt: $R^{-a} \cdot c = p^{-ra} \cdot A^r \cdot m = p^{-ra} \cdot p^{ar} \cdot m = m$

Beispiel

Wie nehmen ein Beispiel:
$p = 47, \quad g = 5$ werden veröffentlicht
B wählt $b = 29$
A wählt $a = 7$
Nachricht $m = 42$

Decisional Diffie-Hellman-Problem (DDH)

Grundgedanke

Angreifer kann zwischen $\langle g^a, g^b, g^{ab} \rangle$ und $\langle g^a, g^b, g^c \rangle$ nicht unterscheiden, wenn a, b und c zufällig gewählt in $[1, |G|]$.

Massey-Omura-Schema

Überblick:

- Das Verfahren stammt von den Kryptologen James Massey und Jim Omura
- Es basiert auf dem diskreten Logarithmus-Problem
- Hauptsächlich zum initialen Tausch von Schlüsseln geeignet

Massey-Omura-Schema

Überblick:

- Das Verfahren stammt von den Kryptologen James Massey und Jim Omura
- Es basiert auf dem diskreten Logarithmus-Problem
- Hauptsächlich zum initialen Tausch von Schlüsseln geeignet

Besonderheit: Es existiert weder ein öffentlicher noch ein gemeinsamer geheimer Schlüssel.

Vorbereitung:

- Wir betrachten die prime Restklassengruppe $(\mathbb{Z}/p\mathbb{Z})^\times$
- Jeder Teilnehmer T wählt ein beliebiges e_T mit:
$$e_T < p - 1 \text{ und } ggT(e_T, p-1) = 1$$
- Bestimme d_T (das multiplikative Inverse) mit:
$$e_T \cdot d_T \equiv 1 \ mod \ (p-1)$$

Für eine Nachricht $m \in (\mathbb{Z}/p\mathbb{Z})^\times$ gilt nun:

$$\begin{aligned}(m^{e_T})^{d_T} &= m^{e_T \cdot d_T} \\ &= m^{k\cdot(p-1)+1} \\ &= m^{k\cdot(p-1)} \cdot m \\ &= m \ mod \ p \qquad (*)\end{aligned}$$

Der Schritt (*) folgt aus dem kleinen Satz von Fermat, da:

$$m^{k\cdot(p-1)} \equiv 1 \ mod \ p$$

Ablauf:

1. **Verschlüsselung Alice:** Alice wählt eine Nachricht $m \in (\mathbb{Z}/p\mathbb{Z})^\times$ und berechnet den Geheimtext:

$$c_1 = m^{e_A} \; mod \; p$$

2. Alice sendet c_1 an Bob.

Ablauf:

1. **Verschlüsselung Alice:** Alice wählt eine Nachricht $m \in (\mathbb{Z}/p\mathbb{Z})^\times$ und berechnet den Geheimtext:

$$c_1 = m^{e_A} \; mod \; p$$

2. Alice sendet c_1 an Bob.
3. **Verschlüsselung Bob:** Bob berechnet den Geheimtext:

$$c_2 = {c_1}^{e_B} \; mod \; p$$

4. Bob sendet c_2 an Alice.

5. **Entschlüsselung Alice:** Alice hebt mit Hilfe von d_A ihre Verschlüsselung auf:

$$\begin{aligned} & c_3 = {c_2}^{d_A} \ mod \ p \\ \Leftrightarrow \ & c_3 = ((m^{e_A})^{e_B})^{d_A} \ mod \ p \\ \Leftrightarrow \ & c_3 = ((m^{e_A})^{d_A})^{e_B} \ mod \ p \\ \Leftrightarrow \ & c_3 = m^{e_B} \ mod \ p \end{aligned}$$

6. Alice sendet c_3 an Bob.

7. **Entschlüsselung Bob:** Bob entschlüsselt die ursprüngliche Nachricht:

$$m = {c_3}^{d_B} \ mod \ p$$
$$\Leftrightarrow m = (m^{e_B})^{d_B} \ mod \ p$$

Ausblick

Peter Shor veröffentlichte 1994 eine Arbeit zum Thema:
Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer